Machine Learning Nanodegree
Capstone Project

“Named Entity Recognition Using CRF”

Project Report

Nitin Bhandari
April 05, 2018

INDEX

1. Definition

1.1 Project Overview:

A ‘named entity’ refers to a word that is not registered in commonly used dictionaries or is a proper noun such as a person’s name, name of a location, or an institution’s name. Named entity recognition refers to the recognition of such named entities within the natural language and categorizes them according to their meaning. “Named Entity Recognition (NER) also known as entity identification, entity chunking, and entity extraction is an important sub task of information extraction that locates and classifies named entities in text into pre-defined categories such as name of the person, location, expressions of time, monetary values, etc.” [1]. It helps to identify the context in which it is written to make sense and extract valuable information from it. But NER possesses serious issues itself. First, there are a huge number of languages and we need annotated data for machines to learn from them. So, quality and quantity of data are one of the key factors to decide the performance of the system. The named entity recognition is a one of the early steps before we can have computers to make sense of the language we speak and write.

For example:

Elon deleted SpaceX page on Facebook on March 23, 2018.

And the system produces an annotated block of text that shows the name of entities:

[Elon]$_{\text{Person}}$ deleted [SpaceX]$_{\text{Organization}}$ page on [Facebook]$_{\text{Organization}}$ on [March 23, 2018]$_{\text{Time}}$

There are three standard approaches to NER:

i. Hand-written regular expressions

ii. Using classifiers:
   - Generative: Naïve Bayes, Perceptron
   - Discriminative: Maxent models

iii. Sequence Models
   - Hidden Markov Models

- Conditional Markov Models / Maximum Entropy Markov Models (CMMs/MEMMs)
- Conditional Random Fields (CRFs)

Related Work:

Much work has been done on Named Entity Recognition in general. Supervised learning is a dominant technique to address this problem. Hai Leong Chieu and Hwee Tou Ng presented a maximum entropy-based name entity recognizer that uses information from the whole document to classify word, with one classifier that uses local context within the sentence as well as uses the context of that word occurring again in the same document to extract useful features to enhance the performance of the named entity recognition. The maximum entropy framework estimates probabilities based on the principle of making as few assumptions as possible, other than the constraints imposed. Such constraints are derived from training data, expressing some relationship between features and outcome. The probability distribution that satisfies the above property is the one with the highest entropy [11].

Other models include Stanford NER that uses CRF models and is implemented in java. “Stanford NER is also known as CRFClassifier and the original CRF code is by Jenny Finkel. Stanford NER is available for download and is licensed under GNU General Public License” [12].

1.2 Problem Statement

NER is a subtask of information extraction. While the expression ‘named entity’ may seem the task to those entities such as word or phrases, stands consistently for a particular reference like those designated as rigid designators but is in fact not strict and can be used for temporal expressions.

For example:

In sentence ‘The automotive company created by Henry Ford in 1903’, Ford refers to the Ford Motor Company although there are other meanings of Ford as well.

‘The year 2001’ and ‘I take my vacations in June’, the first phrase refers to 2001 year of the Georgian calendar and the second phrase refers to the month June of unknown year.

The problem of NER is often broken down into two distinct problems:

a) Detection of named entities (segmentation) and
b) Classification of the name by the type of entity they refer to (Ontology)

The input to the system is the .csv file in which each word in the sentence is annotated with part of speech (POS) tags as shown in figure 1. We use cross-validation and tune the hyper parameters and finally tested on test data and the performance is calculated my looking at the if the system is able to produce the same annotations as given for test data. The output is the accurately identified tags corresponding to the words in the sentence.

1.3 Metrics

The project used F1 score to check the performance of the system. “The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst score at 0. The relative contribution of precision and recall to the F1 score are equal. The formula for the F1 score is:

$$F1 = 2 * (precision * recall) / (precision + recall)$$

Precision is the number of correct positive results divided by the number of all positive results returned by the classifier, and recall is the number of correct positive results divided by all the relevant samples and F1 is the harmonic average of the precision and recall [7].”

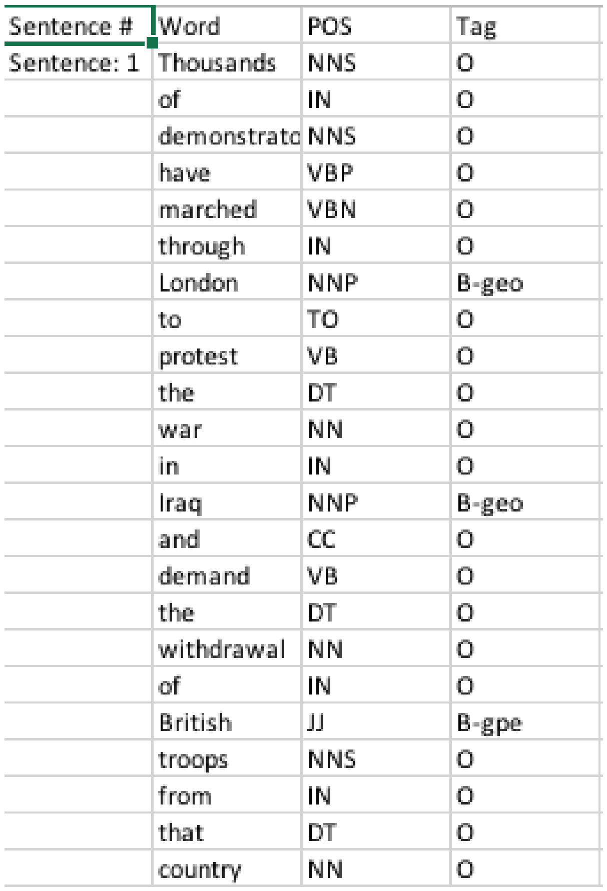
2. Analysis

2.1 Data Exploration

The NLTK python library does not have a standard corpus for NER in English. But it does have the famous CoNLL 2002 Named Entity CoNLL but in Spanish and Dutch. We will be using dataset provided on the Kaggle annotated with Inside, Outside, Beginning (IOB) and POS tags [4]. The dataset can be downloaded from here.
The words tagged with O are outside of named entities. The B-XXX tag is used for the first word in a named entity and I-XXX is used for all other words in named entities of type XXX. The data contains entities of 8 types written below. Following are the essential information about the entities:

- geo = Geographical Entity
- org = Organization
- per = Person
- gpe = Geopolitical Entity
- tim = Time Indicator
- art = Artifact
- eve = Event
- nat = Natural Phenomenon

Target column: 'Tag'
Total word count: 1354149



Sentence #	Word	POS	Tag
Sentence: 1	Thousands	NNS	O
	of	IN	O
	demonstrato	NNS	O
	have	VBP	O
	marched	VBN	O
	through	IN	O
	London	NNP	B-geo
	to	TO	O
	protest	VB	O
	the	DT	O
	war	NN	O
	in	IN	O
	Iraq	NNP	B-geo
	and	CC	O
	demand	VB	O
	the	DT	O
	withdrawal	NN	O
	of	IN	O
	British	JJ	B-gpe
	troops	NNS	O
	from	IN	O
	that	DT	O
	country	NN	O

Fig 1: Sample Input with tags

Figure 1 shows the sample from the NER dataset. The dataset has 4 columns where first column is the sentence, second is the words of the sentence, third is the parts of speech (POS) and fourth is the named entity tag itself. The sentences in the text has been divided into words and each word is annotated. This dataset is unbalanced and therefore F1 is used our evaluation metrics.

2.2 Exploratory Visualization

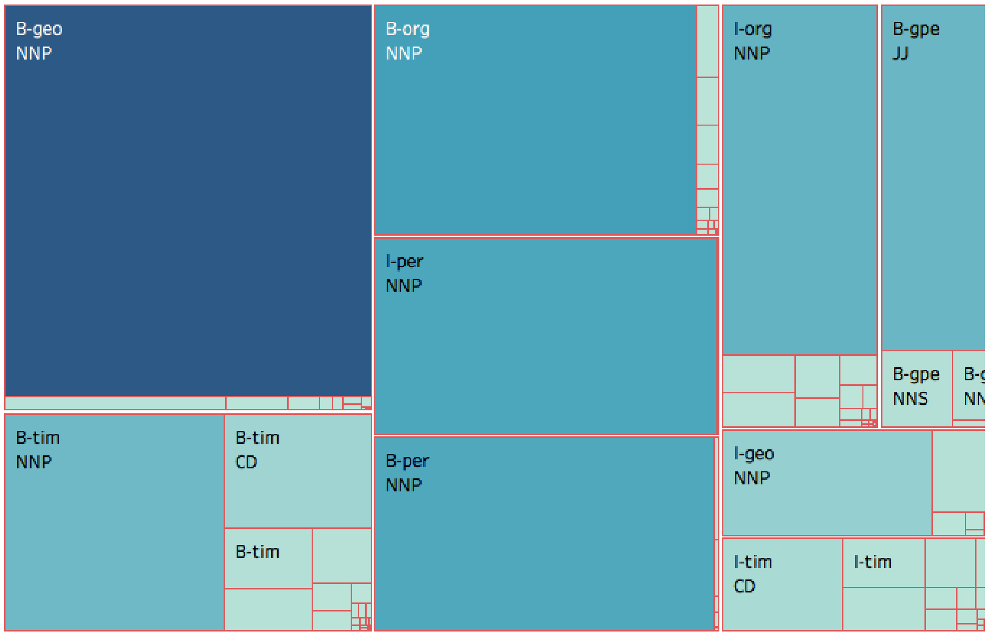


Fig 2: Part of Speech (POS)

Figure 2 shows the different parts of speech (POS) and their frequency in the dataset.

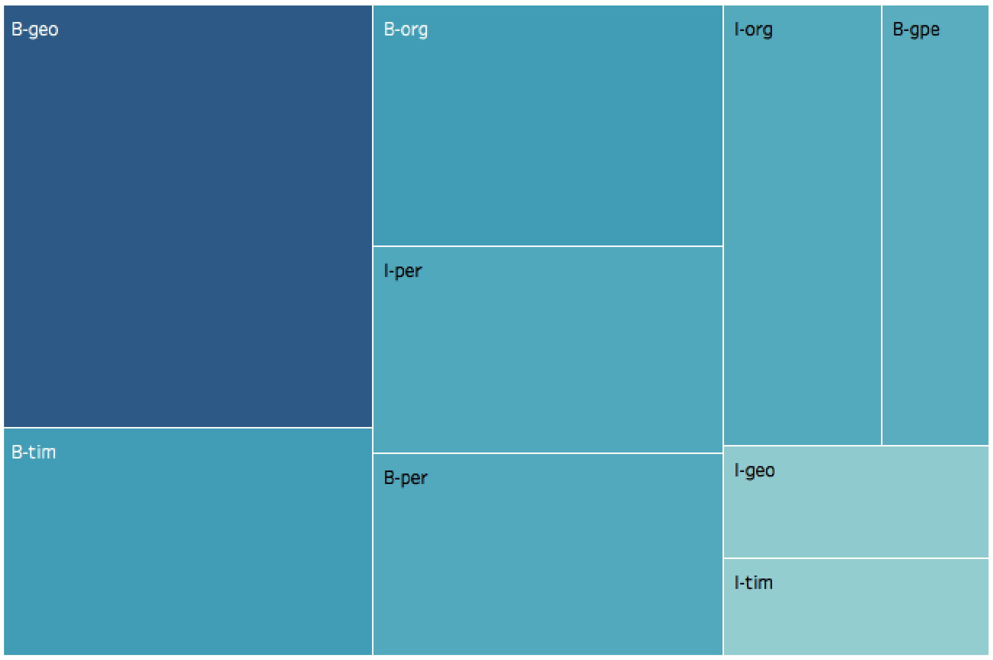


Fig 3: Tags

Figure 3 shows the number of tags and their occurrences in the dataset. Notice that the words like ‘of’, ‘that’, ‘in’ etc. have the tags ‘O’ which are not important since they do not take part in named entity recognition but cannot be removed since we need the context of the sentence not just the words. So, in fig 3 shows all the other tags but not the ‘O’ tag.

2.3 Algorithm and Techniques

Conditional Random Fields (CRF) classifier is used for named entity recognition. The CRF being a class of statistical modelling method which is used in pattern recognition and often used for parsing sequential data in Natural Language Processing(NLP). The CRF model takes the input as the sequence of the words of a sentence and outputs the sequence of named entity tags. In CRF, the conditional probability is modeled by defining the feature map that maps the complete input sequence to a d-dimensional feature vector.

Cross Validation Prediction technique is used that generates cross validation prediction for every input [8].

Perceptron Model is a linear model which is used as a benchmark model and is comparatively simpler than the CRF and is suitable for large scale learning. “By default:

- It does not require a learning rate
- It is not regularized
- It updates its model only on mistakes

The last characteristic implies that the perceptron is slightly faster to train than SGD with the hinge loss and that the resulting model are sparser” [10].

2.4 Benchmark

Perceptron classifier:

To determine the baseline benchmark, we choose perceptron classifier for NER. The reason we choose perceptron model is to see how well CRF model performs as compared to linear model (perceptron). We created a simple feature map and fed it to the perceptron classifier and the perceptron classifier achieved a F1 score of **83%** on the same dataset.

3. Methodology

3.1. Language / Libraries:

Language: Python 2.6

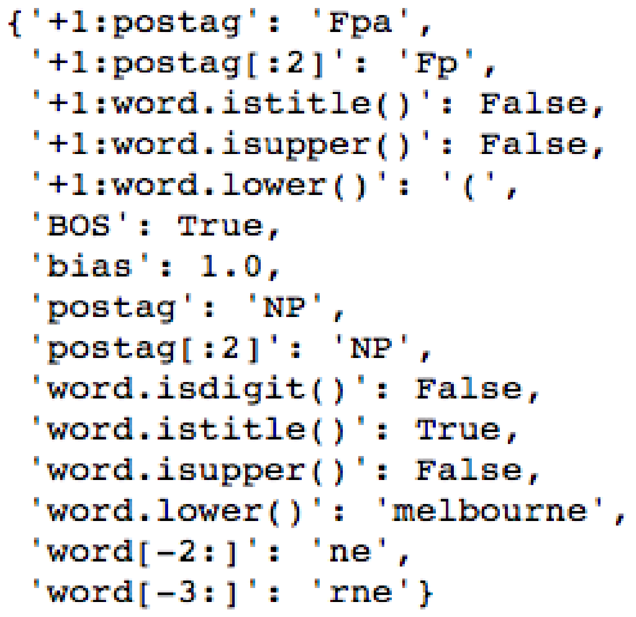
Libraries: pandas, numpy, sklearn

3.2 Data preprocessing

The ***main challenge*** was to find a dataset large enough to generalize the model and is annotated with POS and IOB tags. The null values in the dataset were filled using 'fillna' method and for every word in the sentence, created the tupple (word, POS, Tag) for CRF to work upon. For this feature extractor we used:

- POS tag,
- word identity,
- word suffix
- and some information from neighboring words.

This makes our data ready to be used by CRF. Figure 4 shows the features extracted for the word:



```
{'+1:postag': 'Fpa',
 '+1:postag[:2]': 'Fp',
 '+1:word.istitle()': False,
 '+1:word.isupper()': False,
 '+1:word.lower()': '(',
 'BOS': True,
 'bias': 1.0,
 'postag': 'NP',
 'postag[:2]': 'NP',
 'word.isdigit()': False,
 'word.istitle()': True,
 'word.isupper()': False,
 'word.lower()': 'melbourne',
 'word[-2:]': 'ne',
 'word[-3:]': 'rne'}
```

Fig 4: Features extracted for a word

3.3 Implementation

- First an initial benchmark model was designed using perceptron model and then trained and tested on the same dataset. We created a simple feature vector for every word containing information using POS tags, word identity and target was the tags.

The benchmark model achieved a F1 score of **83%**. The following table gives a clear picture of the code.

Function	Purpose
Feature_map(word)	This function takes in the word and returns the feature vector (a numpy array) using word identity.

Table 1: Functions used in benchmark model

- For the next model, we used Conditional Random Field (CRF) to create a named entity classifier that uses L-BFGS as training algorithm (default) with L1 + L2 regularization. The word_to_feature() function can be made more versatile and include much more information to help classifier learn about the entity and properly name it. This function in the code is the basic implementation of the function. The following table defines the functions used in the CRF model.

Functions	Purpose
get_sentence()	This function creates a list of a list of a sentence with each word in the sentence represented as a tupple of (word, POS, Tag)
Word_to_features(word)	This function along with following 3 supporter-functions are used to extract features from the word using wore identity, POS tags and some information from the nearby words to help the CEF model learn the context of the word.

Table 2: Functions used in CRF model

- We tried to improve the F1 score by tuning the hyper parameters of the CRF model. Hyper parameters to be tuned:
    - C1, which is the coefficient of the L1 regularization, and
    - C2, which is the coefficient of the L2 regularization

  We can try to play with the 'max_iteration' parameter of the CRF model but it turns out to be more time consuming when used with larger values and the score does not got affected after a certain threshold value.

3.4 Refinement

The CRF model shows a significant improvement when compared to benchmark model and achieved F1 score of 0.93.
With tuning of the c1 and c2 hyper parameters of the CRF model, we found the best values for c1 and c2 and model achieved the accuracy of 0.95.

The values of c1 and c2 for the best F1 score are highlighted in table 3 below:

	C1	C2	F1
1	0.1	0.1	0.93
2	**0.3**	**0.1**	**0.95**

Table 3: Hyper parameters of CRF model

4. Result

4.1 Model Evaluation and Validation

The final model outperforms the benchmark and other models used in this Named Entity Recognition (NER) project. The model is evaluated on the basis of their F1 score (performance score) rather than other accuracy scores which can be misleading.

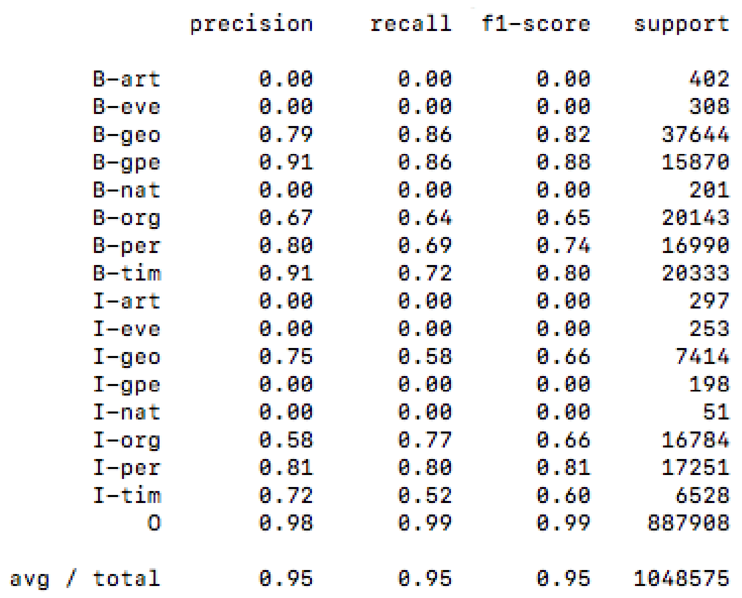
The parameters of the model were carefully tuned to make better predictions. This can be justified by the greater F1 score attained after tuning the hyper parameters.

For this project, Kaggle dataset 'Annotated Corpus for Named Entity recognition' is used which is a CoNLL 2002 dataset annotated with IOB and POS tags and is fairly huge dataset. And 5-fold Cross Validation was used to obtain the performance of all the models used.

Algorithms	F1 Score
Perceptron Model	0.83
CRF Model	0.93
CRF Model (with tuned hyper parameters)	0.95

Table 4: Different models and their F1 score

Sensitivity Analysis:



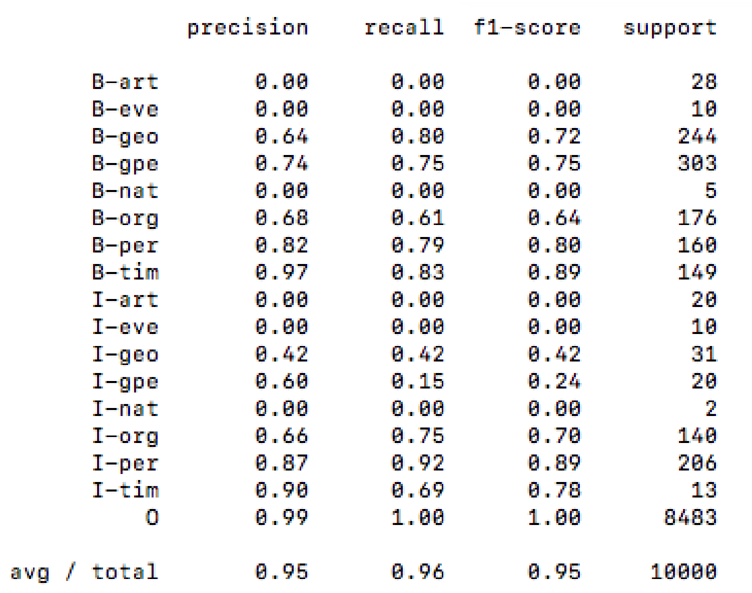
```
             precision    recall  f1-score   support

      B-art       0.00      0.00      0.00       402
      B-eve       0.00      0.00      0.00       308
      B-geo       0.79      0.86      0.82     37644
      B-gpe       0.91      0.86      0.88     15870
      B-nat       0.00      0.00      0.00       201
      B-org       0.67      0.64      0.65     20143
      B-per       0.80      0.69      0.74     16990
      B-tim       0.91      0.72      0.80     20333
      I-art       0.00      0.00      0.00       297
      I-eve       0.00      0.00      0.00       253
      I-geo       0.75      0.58      0.66      7414
      I-gpe       0.00      0.00      0.00       198
      I-nat       0.00      0.00      0.00        51
      I-org       0.58      0.77      0.66     16784
      I-per       0.81      0.80      0.81     17251
      I-tim       0.72      0.52      0.60      6528
          O       0.98      0.99      0.99    887908

avg / total       0.95      0.95      0.95   1048575
```

Fig 5: Cross Validation Report on entire dataset

Figure 5 shows the cross-validation report based on entire dataset and the figure 6 below shows the report on first 10000 samples suggesting that this model is not sensitive to amount of data.



```
             precision    recall  f1-score   support

      B-art       0.00      0.00      0.00        28
      B-eve       0.00      0.00      0.00        10
      B-geo       0.64      0.80      0.72       244
      B-gpe       0.74      0.75      0.75       303
      B-nat       0.00      0.00      0.00         5
      B-org       0.68      0.61      0.64       176
      B-per       0.82      0.79      0.80       160
      B-tim       0.97      0.83      0.89       149
      I-art       0.00      0.00      0.00        20
      I-eve       0.00      0.00      0.00        10
      I-geo       0.42      0.42      0.42        31
      I-gpe       0.60      0.15      0.24        20
      I-nat       0.00      0.00      0.00         2
      I-org       0.66      0.75      0.70       140
      I-per       0.87      0.92      0.89       206
      I-tim       0.90      0.69      0.78        13
          O       0.99      1.00      1.00      8483

avg / total       0.95      0.96      0.95     10000
```

Fig 6: Cross Validation Report on 10000 samples

4.2 Justification

Table 3 shows the top 2 CRF models with their hyper parameter values (c1 and c2). The other models with different value of c1 and c2 drops the F1 score of the model to lower 80%. The model delivers best results on the test data used in 5-fold cross validation (CV). Our goal was to beat the F1 score of the benchmark model i.e. the simple linear model (perceptron), and the CRF model improves the F1 score by 12%.

5. Conclusion

- Free-Form Visualization

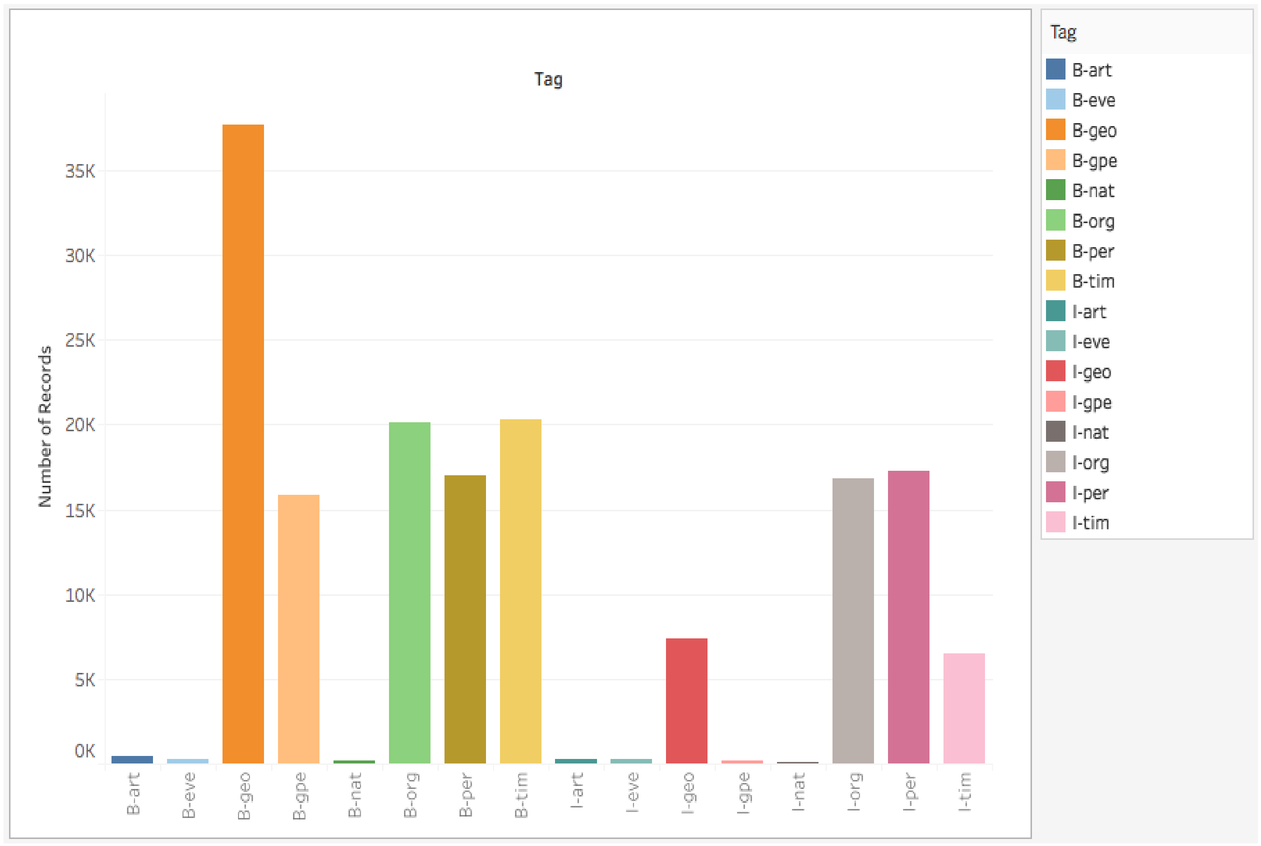


Fig 7: Tag Distribution in data

Figure 7 shows the number of the tags in the entire dataset. The tag ‘O’ that are outside of named entities are not shown. For example, words like ‘the’, ‘of’, ‘in’ etc.

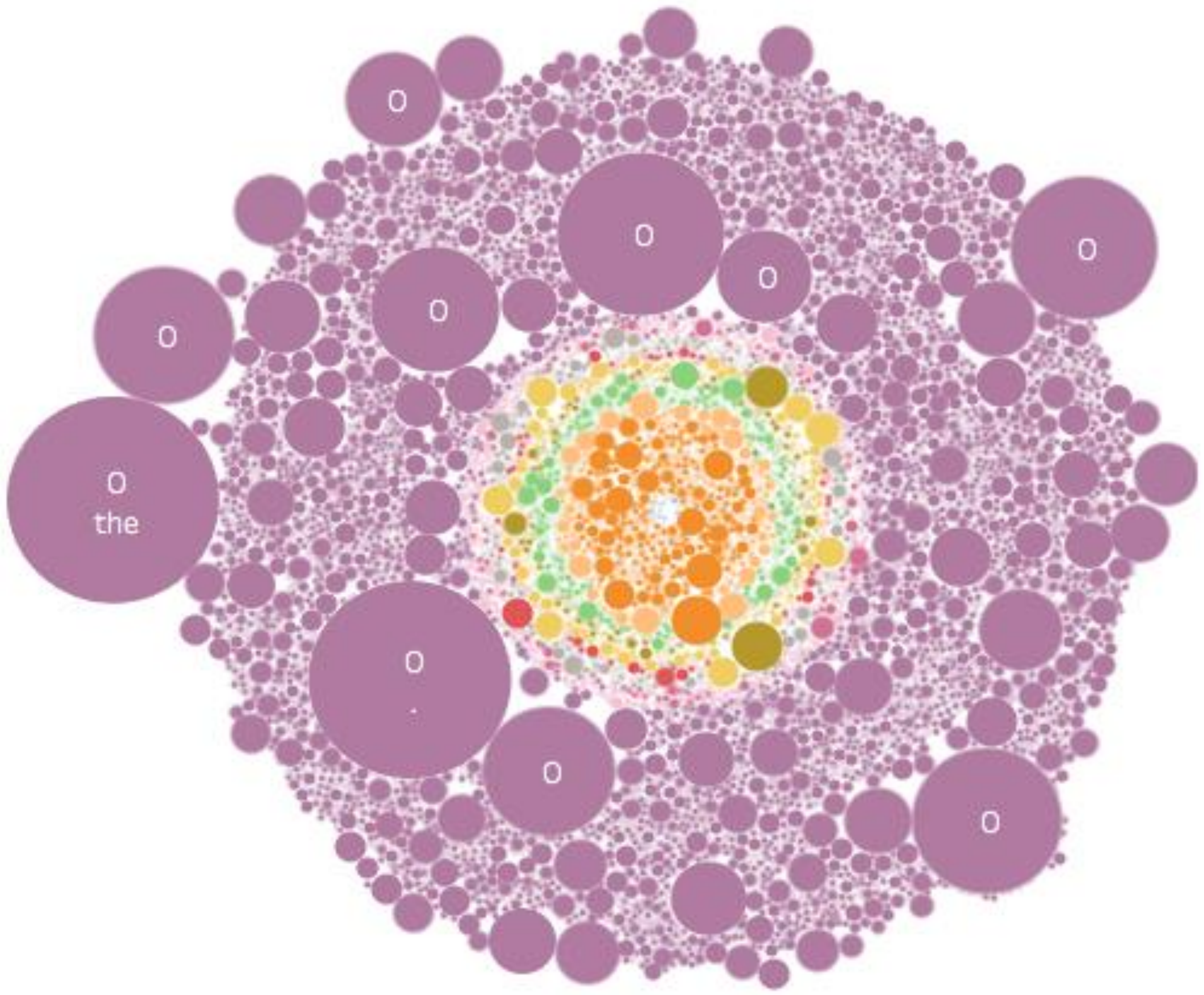


Fig 8: Visualizing tags in the sentences

Figure 8 shows explains the fact that in a sentence, there are only few entities that need to be tagged accordingly. The purple colored tags are ‘O’ tags (outside of named entities) and the rest are the remaining tags that the classifier needs to identify. According to me these ‘O’ tags also play an important role but indirectly to help classifier to classify the named entities not learning the features of the word but learning about the sentence itself which is the reason I decided not to remove these tags from the dataset.

- Reflection

We started from designing benchmark model using perceptron classifier (linear model):

- Read the dataset using pandas
- Made the simple feature vector of 'words' using feature-map function and the target 'tags' and fed it to the perceptron classifier
- Using 5-fold cross validation, the perceptron model was able to achieve a F1 score of 83%

Now, we started building the named entity model using Conditional Random Fields (CRF).

- Read the data and converted the sentence to a tupple in the form of (word, POS, tag)
- Now, we implemented a simple yet efficient feature vector after extracting features using word identity, POS tags, and some information from nearby words
- Finally, it is given as input to the CRF and using 5-fold cross validation achieved and tuning the hyper parameters achieved the F1 score of 95%.

The interesting and challenging part was to implement a feature extractor so that not only the features of the word, but the influence of its nearby words is taken into account and the model tries to learn about the context of the sentence instead of learning about the words.

The final model performs really well in comparison to the benchmark model and I believe it performs satisfactorily and the model seems trustable and aligns well with expected solution outcomes.

- Improvement

Feature Engineering:

Our feature extraction function provides a simple base but can certainly be made more detailed by adding or removing the features.

Data quality and quantity:

The named entity recognition problem relies heavily on the quality and quantity of the data. A good quality data is one with more and more annotated data whereas quantity must be large enough for the system to work correctly on even complex sentences in the language on which it is trained.

Model Selection and tuning:
While CRF are the most common and popular classifiers used for named entity recognition, there are various other models that can be used for named entity recognition task such as bilateral-LSTM's and tuning their hyper parameters.

6. References

[1] https://en.wikipedia.org/wiki/Named-entity_recognition
[2] https://nlpforhackers.io/named-entity-extraction
[3] http://www.iro.umontreal.ca/~lisa/pointeurs/RNNSpokenLanguage2013.pdf
[4] https://www.depends-on-the-definition.com/named-entity-recognition-conditional-random-fields-python
[5] https://en.wikipedia.org/wiki/Conditional_random_field
[6] http://eli5.readthedocs.io/en/latest/tutorials/sklearn_crfsuite.html
[7] http://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html
[8]http://scikitlearn.org/stable/modules/generated/sklearn.model_selection.cross_val_predict.html
[9] https://github.com/TeamHG-Memex/sklearncrfsuite/blob/master/docs/CoNLL2002.ipynb
[10] http://scikit-learn.org/stable/modules/linear_model.html#perceptron
[11] Ki-Joong Lee, Young-Sook Hwang, Seonho Kim, Hae-Chang Rim, "Biomedical named entity recognition using two-phase model based on SVMs", Journal of Biomedical Informatics 37 (2004) 436–447, 22 July 2004
[12] https://nlp.stanford.edu/software/CRF-NER.html